

# Chapter 3

## Components of Hadoop

# Working with files in HDFS

- You can store a big data set of (say) 100TB as a single file in HDFS.
  - Files are replicated and distributed automatically.
  - HDFS abstracts these details away and gives you the illusion that you're dealing with only a single file.

# Basic file commands

- `hadoop fs -cmd <args>`
- Adding files and directories
  - Check the replication factor
- Retrieving files
- Deleting files
- Looking up help

# Reading and writing to HDFS programmatically

- A PutMege program
  - The commandline utilities don't support this operation; we'll use the API.
- In general, Hadoop works more effectively with a single large file rather than a number of small ones.
- The main classes for file manipulation in Hadoop are in package `org.apache.hadoop.fs`

# Anatomy of a MapReduce program

- The general MapReduce data flow. See Figure 3.1
  - Input data is distributed to nodes
  - Each map task works on a “split” of data
  - Mapper outputs intermediate data
  - Data exchange between nodes in a shuffle process
  - Intermediate data of the same key goes to the same reducer
  - Reducer output is stored

# Anatomy of a MapReduce program (Cont'd)

- Hadoop data types
  - Although we can and often do talk about certain keys and values as integers, strings, and so on, they are not exactly standard Java classes, such as Integer, String, and so forth.
  - We need the comparability requirement for keys because they will be sorted at the reduce stage, whereas values are simply passed through.
  - You can create your own custom type. See List 3.2.

# Anatomy of a MapReduce program (Cont'd)

- Mapper
  - Check real codes. The content in the text book is out of date.
- Reducer
  - Check real codes. The content in the text book is out of date.

# Anatomy of a MapReduce program (Cont'd)

- Partitioner
  - Redirecting output from Mapper
  - With multiple reducers, we need some way to determine the appropriate one to send a (key/value) pair outputted by a mapper.
  - The default behavior is to hash the key to determine the reducer.
  - See Figure 3.2 (It's caption).

# Anatomy of a MapReduce program (Cont'd)

- Combiner
  - To perform a “local reduce” before we distribute the mapper results
  - (In Sectin 4.6, page 95) If we specify a combiner, the MapReduce framework may apply it zero, one, or more times to the intermediate data. In order for a combiner to work, it must be an equivalent transformation of the data with respect to the reducer. If we take out the combiner, the reducer’s output will remain the same.